Validating safety and security requirements for partitioned architectures

Julien Delange delange@enst.fr
Laurent Pautet pautet@enst.fr
Peter Feiler phf@sei.cmu.edu

Outline

- Context
- Modeling ARINC653 and MILS systems
- Validating safety and security requirements
- Conclusion

Outline

- Context
- Modeling ARINC653 and MILS systems
- Validating safety and security requirements
- Conclusion

Context and problem

- **Safety-critical systems (avionics, aerospace, …)**
  - Increasing complexity of development
  - Validation/certification needs

- **Dedicated architectures**
  - ARINC653: isolation for safety purposes
  - MILS: isolation for security purposes

- **Detect issues and errors in such architectures**
  - Security and safety policies enforcement
  - Isolation impacts and trade-offs

Proposed approach

- **Partitioned architectures modeling**
  - Appropriate modeling language
  - Dedicated modeling patterns

- **Analysis and validation**
  - Partitioned architecture compliance
  - Requirements validation (is my security policy enforced ?)
  - Trade-off analysis (can safety impacts security ?)

Outline

- Context
- **Modeling ARINC653 and MILS systems**
- Validating safety and security requirements
- Conclusion

Modeling language

- **Modeling requirements**
  - Represent partitioned kernel and partitions runtime
  - Specify runtime requirements (scheduling, safety, …)

- **AADLv2 as a valid candidate**
  - Efficient component approach
  - Appropriate semantics for partitioned systems modeling
  - Extensible, requirements specification

Partitioned architectures modeling

- **Partitioned/isolation kernel modeling**
  - AADL processor
  - Specify isolation requirements (scheduling, memory, etc.)
- **Partition runtime and adress space**
  - AADL virtual processor and process components
  - Model partition communication using AADL ports
- **Hardware association**
  - Memory and partition association (space isolation)
  - Partitions and processor association

Security and safety requirements modeling

- **Define security/safety layers**
  - Define security/criticality levels in virtual bus
  - Use AADL properties (`Criticality_Level` and `Security_Level`)
  - Associates virtual buses to partitions and real buses

- **Specify fault and their containment**
  - Describe error and associated recovery
  - Add relevant information for each level (kernel, partition, thread)

Outline

- Context
- Modeling ARINC653 and MILS systems
- **Validating safety and security requirements**
- Conclusion

Validate architecture consistency

- **Partitioned architecture enforcement**
  - Binding to memory components
  - Correctness of scheduling requirements
- **Memory requirements validation**
  - Space isolation (one memory segment per partition)
  - Memory allocation (has the partition enough memory ?)
- **Scheduling requirements validation**
  - Check time slice correctness (has the partition enough time for its threads?)
  - Ensure partition execution

Validate security policy

- **Consider runtime specificities**
  - Security levels shared inside the partition
  - Inspect security levels and runtime components (bus, etc.)

- **Different security policies, same goal**
  - Biba, Bell-Lapadula: security level domination
  - MILS : security level independence

- **Validation patterns are similar**
  - Connections analysis
  - Hardware analysis (can this hardware support such security level?)

Validate safety policy

- **Check error handling and error recovering**
  - Each level (kernel, partition, thread) handles faults
  - Check recovering correctness (ex: a thread cannot restart the kernel)

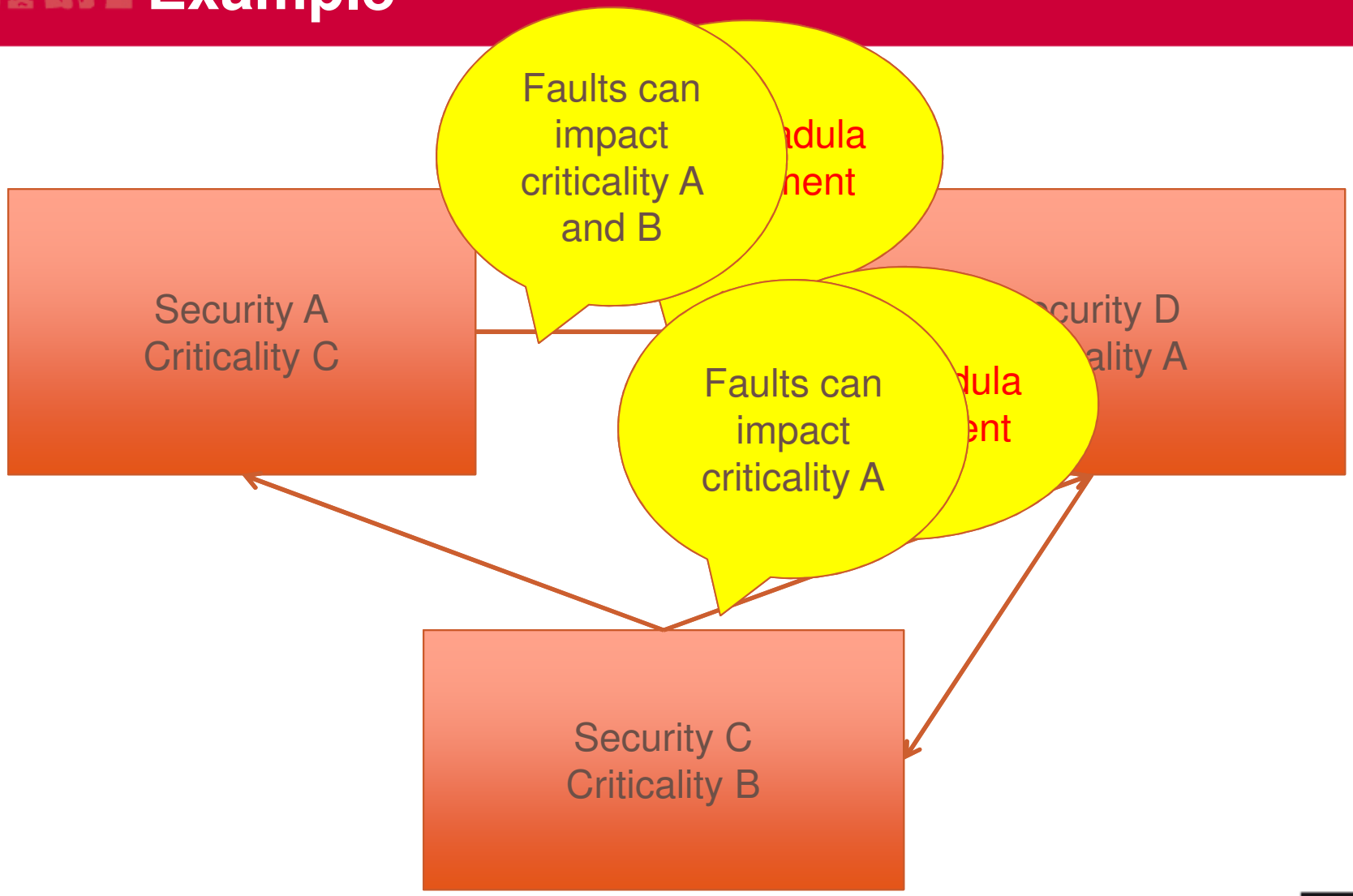
- **Check error path and fault coverage**
  - Recover all faults, anytime
  - Inspect execution paths and their associated errors

Trade-offs analysis

- **Impact of faults and recovering procedures**
  - Impact on communications (omission of data, etc)
  - Impact on runtime system (restart a partition impact several threads)
  - Faults propagation to partitions with higher security/criticality levels
  - Rely on connections flow

- **High-level view, cannot make finer analysis**
  - Need to model application concerns
  - Use the behavior annex

Example


Faults can impact criticality A and B
Security A
Criticality C
Faults can impact criticality A
Security C
Criticality B

Outline

- Context
- Modeling ARINC653 and MILS systems
- Validating safety and security requirements
- Conclusion

Conclusion

- **AADLv2, valid candidate for partitioned systems modeling**
  - Appropriate semantics
  - Security/safety analysis at a high-level
  - Consider runtime requirements and specificities
- **Future work**
  - Automatic implementation from validated models
  - Improve analysis techniques

Thank you for your attention

Note : This work is reflected in the ARINC653 annex of the AADL